SBD PROJECT 2022

ORACLE

-- Table: Booking

CREATE TABLE Booking (

idBooking int NOT NULL,

Guest\_idGuest int NOT NULL,

Room\_idRoom int NOT NULL,

Clerk\_idClerk int NOT NULL,

BookingDate date NOT NULL,

CONSTRAINT Booking\_pk PRIMARY KEY (idBooking)

);

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(0, 4, 4, 0, to\_date('18-Mar-2020','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(1, 0, 1, 1, to\_date('19-Jan-2021','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(2, 1, 2, 3, to\_date('30-Jun-2019','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(3, 3, 3, 2, to\_date('18-May-2019','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(4, 2, 5, 0, to\_date('10-Dec-2021','DD-MON-YYYY'));

-- Table: Caretaker

CREATE TABLE Caretaker (

idCaretaker int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Department\_idDepartment int NOT NULL,

CONSTRAINT Caretaker\_pk PRIMARY KEY (idCaretaker)

);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(0, 'BROWN', 10);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(1, 'PINKMAN', 20);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(2, 'PICASSO', 30);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(3, 'PUTIN', 20);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(4, 'LEE', 10);

-- Table: Clerk

CREATE TABLE Clerk (

idClerk int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Salary int ,

CONSTRAINT Clerk\_pk PRIMARY KEY (idClerk)

);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(0, 'BLAKE', 2300);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(1, 'SMITH', 2100);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(2, 'GREEN', 2000);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(3, 'SHELBY', 3000);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(4, 'LEE', 2560);

-- Table: Department

CREATE TABLE Department (

idDepartment int NOT NULL,

Mngr\_idMngr int NOT NULL,

CONSTRAINT Department\_pk PRIMARY KEY (idDepartment)

);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(10, 2);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(20, 0);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(30, 1);

-- Table: Guest

CREATE TABLE Guest (

idGuest int NOT NULL,

Name nvarchar2(100) NOT NULL,

Surname nvarchar2(100) NOT NULL,

Address nvarchar2(100) NOT NULL,

CONSTRAINT Guest\_pk PRIMARY KEY (idGuest)

);

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(0, 'WALTER', 'WHITE', 'ZLOTA ST. 3');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(1, 'BRUCE', 'WAYNE', 'WEST ST. 41');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(2, 'RICK', 'SANCHEZ', 'AZADLIQ ST. 26');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(3, 'ABUSAT', 'AGHALI', 'M.ALIYEV ST. 130');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(4, 'ELON', 'MUSK', 'EAST ST. 113');

-- Table: Manager

CREATE TABLE Mngr (

idMngr int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Salary int NOT NULL,

CONSTRAINT Mngr\_pk PRIMARY KEY (idMngr)

);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (0, 'JACKSON', 20000);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (1, 'TRUMAN', 15000);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (2, 'FLOYD', 25500);

-- Table: Room

CREATE TABLE Room (

idRoom int NOT NULL,

Beds int NOT NULL,

Price int NOT NULL,

Caretaker\_idCaretaker int NOT NULL,

CONSTRAINT Room\_pk PRIMARY KEY (idRoom)

);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (1, 3, 500, 4);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (2, 3, 500, 1);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (3, 2, 350, 2);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (4, 3, 500, 0);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (5, 1, 100, 4);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (6, 1, 90, 3);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (7, 2, 300, 1);

-- foreign keys

-- Reference: Booking\_Clerk (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Clerk

FOREIGN KEY (Clerk\_idClerk)

REFERENCES Clerk (idClerk);

-- Reference: Booking\_Guest (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Guest

FOREIGN KEY (Guest\_idGuest)

REFERENCES Guest (idGuest);

-- Reference: Booking\_Room (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Room

FOREIGN KEY (Room\_idRoom)

REFERENCES Room (idRoom);

-- Reference: Department\_Manager (table: Department)

ALTER TABLE Department ADD CONSTRAINT Department\_Mngr

FOREIGN KEY (Mngr\_idMngr)

REFERENCES Mngr (idMngr);

-- Reference: Room\_Caretaker (table: Room)

ALTER TABLE Room ADD CONSTRAINT Room\_Caretaker

FOREIGN KEY (Caretaker\_idCaretaker)

REFERENCES Caretaker (idCaretaker);

-- Reference: Staff\_Department (table: Caretaker)

ALTER TABLE Caretaker ADD CONSTRAINT Staff\_Department

FOREIGN KEY (Department\_idDepartment)

REFERENCES Department (idDepartment);

-- PROCEDURES

--Increase price for 3-beds rooms by 10% and for 2-beds rooms by 15%

CREATE OR REPLACE PROCEDURE IncreasePrice

As

NoRoom Exception;

counting Number;

idR Number;

CURSOR rm IS

SELECT idRoom

FROM Room

WHERE beds = 2 OR beds = 3;

Begin

Select COUNT(idRoom) into counting

From Room

Where beds = 3 or beds = 2;

IF counting = 0

Then raise NoRoom;

ELSE

OPEN rm;

LOOP

FETCH rm into idR;

EXIT WHEN rm%notfound;

Update Room

Set price = price\*1.15

Where beds = 2 and idRoom = idR;

Update Room

Set price = price\*1.1

Where beds = 3 and idRoom = idR;

End Loop;

END IF;

Exception When NoRoom Then dbms\_output.put\_line('There are no such rooms');

END;

select idRoom, price

From Room

Where beds = 3;

Exec IncreasePrice;

--Add bonus to the salary of clerk(s) with given surname by given value

CREATE OR REPLACE PROCEDURE RaiseSal(cName nvarchar2, bonus int)

As

NoClerk Exception;

CURSOR person IS

Select idClerk

From Clerk

Where surname = cName;

counting Number;

idP Number;

Begin

Select COUNT(idClerk) into counting

From Clerk

Where surname = cName;

IF counting = 0 Then raise NoClerk;

Else

Open person;

Loop

Fetch person into idP;

Exit When person%notfound;

Update Clerk

Set Salary = Salary+bonus

Where idClerk = idP;

End Loop;

dbms\_output.put\_line('Salary of clerk ' || cName || ' raised by' || bonus);

END IF;

Exception When NoClerk Then

dbms\_output.put\_line('There are no such clerk');

END;

select surname, salary

from clerk

where surname = 'BLAKE';

Exec RaiseSal('BLAKE',200);

-- Add new booking for the guest with given id to the given room which its clerk will be the one with highest salary.

CREATE OR REPLACE PROCEDURE AddBooking(idPerson int, idR int)

As

NoGuest Exception;

CURSOR clerk IS

Select idClerk

From Clerk

Where salary in (Select MAX(salary) from clerk);

countG Number;

countR Number;

idNewBooking Number;

idC Number;

Begin

Select COUNT(idGuest) into countG

from Guest

Where idGuest = idPerson;

IF countG = 0 Then raise NoGuest;

ELSE

Open clerk;

Loop

Fetch clerk into idC;

Exit When clerk%notfound;

Select NVL(MAX(idBooking)+1,1) into idNewBooking

From Booking;

INSERT INTO Booking VALUES(idNewBooking,idPerson,idR,idC,sysdate);

End Loop;

END IF;

Exception When NoGuest Then dbms\_output.put\_line('No such guest');

END;

Select idClerk from Clerk where salary in(select max(salary) from clerk);

select \*

from Booking;

Exec AddBooking(4,6);

--TRIGGERS

-- Trigger which is enabled when removing a room which is booked at the moment.

create or replace trigger remove\_active\_room\_trig

before delete

on Room

for each row

declare

errorMessage Exception;

cursor a1 is

select \* from Booking;

begin

for bk in a1 loop

if bk.Room\_idRoom = :OLD.idRoom and bk.BookingDate < sysdate

then raise errorMessage;

end if;

end loop;

Exception when errorMessage then

RAISE\_APPLICATION\_ERROR(-20000,'Cannot remove, the room is in use at the moment.');

end;

select \*

from booking;

delete from Room

where idRoom = 4;

-- Trigger which is enabled when creating a booking on a room that is already booked at the moment.

create or replace trigger book\_for\_same\_day\_trig

before insert

on Booking

for each row

declare

errorMessage Exception;

x number;

currentDate date;

cursor a1 is

select \* from Room;

begin

currentDate := :new.BookingDate;

select count(idBooking) into x

from Booking

where BookingDate = currentDate;

for rm in a1 loop

if :new.Room\_idRoom = rm.idRoom and x > 0

then raise errorMessage;

end if;

end loop;

Exception when errorMessage then

RAISE\_APPLICATION\_ERROR(-20000, 'Room has already been scheduled for the same date.');

end;

/

select \* from booking;

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(6, 2, 1, 0, to\_date('10-Dec-2021','DD-MON-YYYY'));

--Trigger which raises the manager's salary by 10% of some department which a new caretaker is inserted.

create or replace trigger raise\_mgr\_sal\_trig

after insert on Caretaker

for each row

declare

cursor m1 is

select idMngr from Mngr;

currentID int;

idM int;

idD int;

begin

open m1;

idD := :new.Department\_idDepartment;

Select Mngr\_idMngr into idM

from Department

Where idDepartment = idD;

loop

fetch m1 into currentID;

exit when m1%notfound;

if currentID = idM

then

Update Mngr

Set Salary = Salary\*1.1

Where idMngr = idM;

DBMS\_OUTPUT.PUT\_LINE('Salary of manager with id ' || idM || ' raised by 10%.');

end if;

end loop;

close m1;

end;

select idMngr, Salary

From Mngr;

Select idDepartment,Mngr\_idMngr

from Department

Where Mngr\_idMngr = 1;

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(5, 'DRAKE', 30);

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(0, 4, 4, 0, to\_date('18-Mar-2020','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(1, 0, 1, 1, to\_date('19-Jan-2021','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(2, 1, 2, 3, to\_date('30-Jun-2019','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(3, 3, 3, 2, to\_date('18-May-2019','DD-MON-YYYY'));

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(4, 2, 0, 0, to\_date('10-Dec-2021','DD-MON-YYYY'));

-- Table: Caretaker

CREATE TABLE Caretaker (

idCaretaker int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Department\_idDepartment int NOT NULL,

CONSTRAINT Caretaker\_pk PRIMARY KEY (idCaretaker)

);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(0, 'BROWN', 10);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(1, 'PINKMAN', 20);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(2, 'PICASSO', 30);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(3, 'PUTIN', 20);

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(4, 'LEE', 10);

-- Table: Clerk

CREATE TABLE Clerk (

idClerk int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Salary int ,

CONSTRAINT Clerk\_pk PRIMARY KEY (idClerk)

);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(0, 'BLAKE', 2300);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(1, 'SMITH', 2100);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(2, 'GREEN', 2000);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(3, 'SHELBY', 3000);

INSERT INTO Clerk(idClerk, Surname, Salary) VALUES(4, 'LEE', 2560);

-- Table: Department

CREATE TABLE Department (

idDepartment int NOT NULL,

Mngr\_idMngr int NOT NULL,

CONSTRAINT Department\_pk PRIMARY KEY (idDepartment)

);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(10, 2);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(20, 0);

INSERT INTO Department(idDepartment, Mngr\_idMngr) VALUES(30, 1);

-- Table: Guest

CREATE TABLE Guest (

idGuest int NOT NULL,

Name nvarchar2(100) NOT NULL,

Surname nvarchar2(100) NOT NULL,

Address nvarchar2(100) NOT NULL,

CONSTRAINT Guest\_pk PRIMARY KEY (idGuest)

);

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(0, 'WALTER', 'WHITE', 'ZLOTA ST. 3');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(1, 'BRUCE', 'WAYNE', 'WEST ST. 41');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(2, 'RICK', 'SANCHEZ', 'AZADLIQ ST. 26');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(3, 'ABUSAT', 'AGHALI', 'M.ALIYEV ST. 130');

INSERT INTO Guest (idGuest, Name, Surname, Address)

VALUES(4, 'ELON', 'MUSK', 'EAST ST. 113');

-- Table: Manager

CREATE TABLE Mngr (

idMngr int NOT NULL,

Surname nvarchar2(100) NOT NULL,

Salary int NOT NULL,

CONSTRAINT Mngr\_pk PRIMARY KEY (idMngr)

);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (0, 'JACKSON', 20000);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (1, 'TRUMAN', 15000);

INSERT INTO Mngr (idMngr, Surname, Salary) VALUES (2, 'FLOYD', 25500);

-- Table: Room

CREATE TABLE Room (

idRoom int NOT NULL,

Beds int NOT NULL,

Price int NOT NULL,

Caretaker\_idCaretaker int NOT NULL,

CONSTRAINT Room\_pk PRIMARY KEY (idRoom)

);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (1, 3, 500, 4);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (2, 3, 500, 1);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (3, 2, 350, 2);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (4, 3, 500, 0);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (5, 1, 100, 4);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (6, 1, 90, 3);

INSERT INTO Room (idRoom, Beds, Price, Caretaker\_idCaretaker) VALUES (7, 2, 300, 1);

-- foreign keys

-- Reference: Booking\_Clerk (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Clerk

FOREIGN KEY (Clerk\_idClerk)

REFERENCES Clerk (idClerk);

-- Reference: Booking\_Guest (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Guest

FOREIGN KEY (Guest\_idGuest)

REFERENCES Guest (idGuest);

-- Reference: Booking\_Room (table: Booking)

ALTER TABLE Booking ADD CONSTRAINT Booking\_Room

FOREIGN KEY (Room\_idRoom)

REFERENCES Room (idRoom);

-- Reference: Department\_Manager (table: Department)

ALTER TABLE Department ADD CONSTRAINT Department\_Mngr

FOREIGN KEY (Mngr\_idMngr)

REFERENCES Mngr (idMngr);

-- Reference: Room\_Caretaker (table: Room)

ALTER TABLE Room ADD CONSTRAINT Room\_Caretaker

FOREIGN KEY (Caretaker\_idCaretaker)

REFERENCES Caretaker (idCaretaker);

-- Reference: Staff\_Department (table: Caretaker)

ALTER TABLE Caretaker ADD CONSTRAINT Staff\_Department

FOREIGN KEY (Department\_idDepartment)

REFERENCES Department (idDepartment);

-- PROCEDURES

--Increase price for 3-beds rooms by 10% and for 2-beds rooms by 15%

CREATE OR REPLACE PROCEDURE IncreasePrice

As

NoRoom Exception;

counting Number;

idR Number;

CURSOR rm IS

SELECT idRoom

FROM Room

WHERE beds = 2 OR beds = 3;

Begin

Select COUNT(idRoom) into counting

From Room

Where beds = 3 or beds = 2;

IF counting = 0

Then raise NoRoom;

ELSE

OPEN rm

LOOP

FETCH rm into idR;

EXIT WHEN rm%notfound;

Update Room

Set price = price\*1.15

Where beds = 2 and idRoom = idR;

Update Room

Set price = price\*1.1

Where beds = 3 and idRoom = idR;

End Loop;

END IF;

Exception When NoRoom Then dbms\_output.put\_line('There are no such rooms');

END;

select idRoom, price

From Room

Where beds = 3;

Exec IncreasePrice;

--Add bonus to the salary of clerk with given surname by given value

CREATE OR REPLACE PROCEDURE RaiseSal(cName nvarchar2, bonus int)

As

NoClerk Exception;

CURSOR person IS

Select idClerk

From Clerk

Where surname = cName;

counting Number;

idP Number;

Begin

Select COUNT(idClerk) into counting

From Clerk

Where surname = cName;

IF counting = 0 Then raise NoClerk;

Else

Open person;

Loop

Fetch person into idP;

Exit When person%notfound;

Update Clerk

Set Salary = Salary+bonus

Where idClerk = idP;

End Loop;

dbms\_output.put\_line('Salary of clerk ' || cName || ' raised by' || bonus);

END IF;

Exception When NoClerk Then

dbms\_output.put\_line('There are no such clerk');

END;

select surname, salary

from clerk

where surname = 'BLAKE';

Exec RaiseSal('BLAKE',200);

-- Add new booking for the guest with given id to the given room which its clerk will be the one with highest salary.

CREATE OR REPLACE PROCEDURE AddBooking(idPerson int, idR int)

As

NoGuest Exception;

CURSOR clerk IS

Select idClerk

From Clerk

Where salary in (Select MAX(salary) from clerk);

countG Number;

countR Number;

idNewBooking Number;

idC Number;

Begin

Select COUNT(idGuest) into countG

from Guest

Where idGuest = idPerson;

IF countG = 0 Then raise NoGuest;

ELSE

Open clerk;

Loop

Fetch clerk into idC;

Exit When clerk%notfound;

Select NVL(MAX(idBooking)+1,1) into idNewBooking

From Booking;

INSERT INTO Booking VALUES(idNewBooking,idPerson,idR,idC,sysdate);

End Loop;

END IF;

Exception When NoGuest Then dbms\_output.put\_line('No such guest');

END;

select \*

from Booking;

Exec AddBooking(4,6);

--TRIGGERS

-- Trigger which is enabled when removing a room which is booked at the moment.

create or replace trigger remove\_active\_room\_trig

before delete

on Room

for each row

declare

errorMessage Exception;

cursor a1 is

select \* from Booking;

begin

for bk in a1 loop

if bk.Room\_idRoom = :OLD.idRoom and bk.BookingDate < sysdate

then raise errorMessage;

end if;

end loop;

Exception when errorMessage then

RAISE\_APPLICATION\_ERROR(-20000,'Cannot remove, the room is in use at the moment.');

end;

select \*

from booking;

delete from Room

where idRoom = 4;

-- Trigger which is enabled when creating a booking on a room that is already booked at the moment.

create or replace trigger book\_for\_same\_day\_trig

before insert

on Booking

for each row

declare

errorMessage Exception;

x number;

currentDate date;

cursor a1 is

select \* from Room;

begin

currentDate := :new.BookingDate;

select count(idBooking) into x

from Booking

where BookingDate = currentDate;

for rm in a1 loop

if :new.Room\_idRoom = rm.idRoom and x > 0

then raise errorMessage;

end if;

end loop;

Exception when errorMessage then

RAISE\_APPLICATION\_ERROR(-20000, 'Room number ' || rm.idRoom || ' has already been scheduled for the same date.');

end;

/

select \* from booking;

INSERT INTO Booking(idBooking, Guest\_idGuest, Room\_idRoom, Clerk\_idClerk, BookingDate)

VALUES(6, 2, 1, 0, to\_date('10-Dec-2021','DD-MON-YYYY'));

--Trigger which raises the manager's salary by 10% of some department which a new caretaker is inserted.

create or replace trigger raise\_mgr\_sal\_trig

after insert on Caretaker

for each row

declare

cursor m1 is

select idMngr from Mngr;

currentID int;

idM int;

idD int;

begin

open m1;

idD := :new.Department\_idDepartment;

Select Mngr\_idMngr into idM

from Department

Where idDepartment = idD;

loop

fetch m1 into currentID;

exit when m1%notfound;

if currentID = idM

then

Update Mngr

Set Salary = Salary\*1.1

Where idMngr = idM;

DBMS\_OUTPUT.PUT\_LINE('Salary of manager with id ' || idM || ' raised by 10%.');

end if;

end loop;

close m1;

end;

select idMngr, Salary

From Mngr;

Select idDepartment,Mngr\_idMngr

from Department

Where Mngr\_idMngr = 1;

INSERT INTO Caretaker(idCaretaker, Surname, Department\_idDepartment)

VALUES(5, 'DRAKE', 30);

T SQL

use[s21460];

--Table Booking

CREATE TABLE Booking(booking\_no INT not null , guest\_id INT not null ,

room\_id INT not null , clerk\_id INT not null , booking\_date date not null ,

constraint GUEST\_FK foreign key (guest\_id) references Guest(guest\_id),

constraint ROOM\_FK foreign key (room\_id) references Room(room\_id),

constraint CLERK\_FK foreign key (clerk\_id) references Guest(guest\_id),

constraint BOOK\_PRIMARY\_KEY primary key (booking\_no));

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(0, 4, 4, 0, '18-Mar-2020');

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(1, 0, 1, 1, '19-Jan-2021');

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(2, 1, 2, 3, '30-Jun-2019');

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(3, 3, 3, 2, '18-May-2019');

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(4, 2, 5, 0, '10-Dec-2021');

--Table Caretaker

CREATE TABLE Caretaker(caretaker\_id INT not null , surname char(14) not null ,

department\_id INT not null ,

constraint DEP\_FK foreign key (department\_id) references Department(department\_id),

constraint CARETAKER\_PK primary key (caretaker\_id));

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(0, 'BROWN', 10);

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(1, 'PINKMAN', 20);

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(2, 'PICASSO', 30);

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(3, 'PUTIN', 20);

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(4, 'LEE', 10);

--Table Clerk

CREATE TABLE Clerk(clerk\_id INT not null , surname char(14) not null ,

salary INT,

constraint CLERK\_PK primary key (clerk\_id));

INSERT INTO Clerk(clerk\_id, surname, salary) VALUES(0, 'BLAKE', 2300);

INSERT INTO Clerk(clerk\_id, surname, salary) VALUES(1, 'SMITH', 2100);

INSERT INTO Clerk(clerk\_id, surname, salary) VALUES(2, 'GREEN', 2000);

INSERT INTO Clerk(clerk\_id, surname, salary) VALUES(3, 'SHELBY', 3000);

INSERT INTO Clerk(clerk\_id, surname, salary) VALUES(4, 'LEE', 2560);

--Table Department

CREATE TABLE Department(department\_id INT not null , manager\_id INT not null ,

constraint MGR\_FK foreign key (manager\_id) references Manager(manager\_id),

constraint DEPARTMENT\_PK primary key (department\_id));

INSERT INTO Department(department\_id, manager\_id) VALUES(10, 2);

INSERT INTO Department(department\_id, manager\_id) VALUES(20, 0);

INSERT INTO Department(department\_id, manager\_id) VALUES(30, 1);

--Table Guest

CREATE TABLE Guest(guest\_id INT not null , name char(14) not null ,

surname char(14) not null ,

constraint GUEST\_PK primary key (guest\_id));

INSERT INTO Guest (guest\_id, name, surname)

VALUES(0, 'WALTER', 'WHITE');

INSERT INTO Guest (guest\_id, name, surname)

VALUES(1, 'BRUCE', 'WAYNE');

INSERT INTO Guest (guest\_id, name, surname)

VALUES(2, 'RICK', 'SANCHEZ');

INSERT INTO Guest (guest\_id, name, surname)

VALUES(3, 'ABUSAT', 'AGHALI');

INSERT INTO Guest (guest\_id, name, surname)

VALUES(4, 'ELON', 'MUSK');

--Table Manager

CREATE TABLE Manager(manager\_id INT not null , surname char(14) not null ,

salary INT,

constraint MANAGER\_PK primary key (manager\_id));

INSERT INTO Manager (manager\_id, surname, salary) VALUES (0, 'JACKSON', 20000);

INSERT INTO Manager (manager\_id, surname, salary) VALUES (1, 'TRUMAN', 15000);

INSERT INTO Manager (manager\_id, surname, salary) VALUES (2, 'FLOYD', 25500);

--Table Room

CREATE TABLE Room(room\_id INT not null , beds INT not null ,

price numeric(4), caretaker\_id INT not null ,

constraint CARETAKER\_FK foreign key (caretaker\_id) references Caretaker(caretaker\_id),

constraint ROOM\_PK primary key (room\_id));

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (1, 3, 500, 4);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (2, 3, 500, 1);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (3, 2, 350, 2);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (4, 3, 500, 0);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (5, 1, 100, 4);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (6, 1, 90, 3);

INSERT INTO Room (room\_id, beds, price, caretaker\_id) VALUES (7, 2, 300, 1);

--PROCEDURES

--Reduce price for 3-beds rooms by 20% and for 2-beds rooms by 10% (RESULT SET PROCEDURE)

CREATE PROCEDURE ReducePrice

As

Begin

Declare @idR INT;

Declare rm CURSOR LOCAL FOR

Select room\_id

from Room

Where beds = 2 or beds = 3;

Declare @counting INT;

Select @counting = COUNT(room\_id)

From Room

Where beds = 2 or beds = 3;

Open rm;

While 1=1 begin

Fetch rm into @idR;

If @@FETCH\_STATUS <> 0 BREAK ;

Update Room

Set price = price\*0.9

Where beds = 2 and room\_id = @idR;

Update Room

Set price = price\*0.8

Where beds = 3 and room\_id = @idR;

End;

close rm;

print 'Prices reduced for 2-bed rooms by 10% and for 3-bed rooms by 20%.';

Select room\_id, beds, price

from Room

Where beds = 2 or beds = 3

END; ----

Select room\_id, beds, price

from Room

Where beds = 2 or beds = 3

Exec ReducePrice;

--Add bonus to the salary of clerk with given surname by given value and return the new salary.(RETURN PROCEDURE)

CREATE PROCEDURE RaiseSal(@cName nvarchar(2000), @bonus int)

As

Begin

DECLARE person CURSOR LOCAL FOR

Select clerk\_id

From Clerk

Where surname = @cName;

Declare @counting INT;

Declare @idP INT;

Declare @retSal INT;

Select @counting = COUNT(clerk\_id)

From Clerk

Where surname = @cName;

Open person;

While 1=1 begin

Fetch person into @idP;

If @@FETCH\_STATUS <> 0 BREAK ;

Update Clerk

Set Salary = Salary+@bonus

Where clerk\_id = @idP;

End;

Select @retSal = salary

from Clerk

Where clerk\_id = @idP;

return @retSal;

END; ---

select salary

from Clerk

Where surname = 'LEE';

Declare @x int;

Execute @x = RaiseSal 'LEE',300;

print @x;

----

-- Add new booking for the guest with given id to the given room which its clerk will be the one with highest salary.

-- (OUTPUT PROCEDURE)

CREATE PROCEDURE AddBooking

@idPerson int, @idR int,

@message varchar OUTPUT

As

Begin

DECLARE clerk CURSOR LOCAL FOR

Select clerk\_id

From Clerk

Where salary = (Select MAX(salary) from clerk);

Declare @countG INT;

Declare @idNewBooking INT;

Declare @idC INT;

Select @countG = COUNT(guest\_id)

from Guest

Where guest\_id = @idPerson;

Open clerk;

While 1=1 begin

Fetch clerk into @idC;

If @@FETCH\_STATUS <> 0 BREAK ;

Select @idNewBooking = ISNULL(MAX(booking\_no)+1,1)

From Booking;

INSERT INTO Booking VALUES(@idNewBooking,@idPerson,@idR,@idC,GETDATE());

End;

Select @message = 'Guest with id ' + CAST(@idPerson as varchar) + ' added new booking(id ' + CAST(@idNewBooking as varchar) +

') to room number ' + CAST(@idR as varchar);

END; ------

drop procedure AddBooking;

select clerk\_id

from Clerk

Where salary = (select MAX(salary) from Clerk);

Declare @m varchar;

Execute AddBooking 4,6,@m OUTPUT;

Print @m;

----

--TRIGGERS

-- Trigger which is enabled when removing a room which is booked at the moment.

create trigger remove\_active\_room\_trig on Room for delete as

declare @old\_id INT;

select @old\_id = room\_id from deleted;

declare b1 cursor local for

select room\_id from Booking;

declare @tempId INT;

declare @old\_date date;

select @old\_date = booking\_date from booking where room\_id = @old\_id;

begin

open b1;

while 1=1 begin

fetch b1 into @tempId;

If @@FETCH\_STATUS <> 0 BREAK ;

if @tempId = @old\_id and @old\_date < GETDATE()

THROW 51000, 'Cannot remove, the room is in use at the moment.', 1;

End;

close b1;

END;

select \*

from booking;

delete from Room

Where room\_id = 2;

-----

-- Trigger which is enabled when creating a booking on a room that is already booked at the moment.

create trigger book\_for\_same\_date on Booking for insert as

declare @x INT;

declare @currDate date;

select @currDate = booking\_date from inserted;

declare @new\_roomid INT;

select @new\_roomid = room\_id from inserted;

declare @tempId INT;

declare r1 CURSOR LOCAL FOR

select room\_id from Room;

select @x = COUNT(booking\_no)

from Booking

Where booking\_date = @currDate;

begin

open r1;

while 1=1 begin

fetch r1 into @tempId;

If @@FETCH\_STATUS <> 0 BREAK ;

if @new\_roomid = @tempId and @x > 0

THROW 51000,'This room has already been scheduled for the same date.', 1;

End;

close r1;

END;

select \* from booking;

INSERT INTO Booking(booking\_no, guest\_id, room\_id, clerk\_id, booking\_date)

VALUES(5, 2, 5, 0, '10-Dec-2021');

----

--Trigger which raises the manager's salary by 10% of some department which a new caretaker is inserted.

create trigger raise\_mgr\_sal\_trig on Caretaker for insert as

declare m1 CURSOR LOCAL FOR

select manager\_id from Manager;

declare @currId INT;

declare @idM INT;

declare @idD INT;

select @idD = department\_id from inserted;

select @idM = manager\_id from Department where department\_id = @idD;

begin

open m1;

while 1=1 begin

fetch m1 into @currId;

If @@FETCH\_STATUS <> 0 BREAK ;

if @currId = @idM

Update Manager

Set salary = salary\*1.1

where manager\_id = @idM;

print 'Salary of manager with id ' + CAST(@idM as varchar) + ' raised by 10%.';

End;

close m1;

END;

select manager\_id, salary

From Manager;

Select department\_id,manager\_id

from Department

Where manager\_id = 1;

INSERT INTO Caretaker(caretaker\_id, surname, department\_id)

VALUES(5, 'DRAKE', 30);